Distributed Node Coordination for Real-Time Energy-Constrained Control in Wireless Sensor and Actuator Networks

Lei Mo, Xianghui Cao, Ye-Qiong Song, Angeliki Kritikakou

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/JIOT.2018.2839030, IEEE Internet of Things Journal

Distributed Node Coordination for Real-Time Energy-Constrained Control in Wireless Sensor and Actuator Networks

Lei Mo*, *Member, IEEE,* Xianghui Cao†, *Senior Member, IEEE,* Yeqiong Song‡, *Member, IEEE,* and Angeliki Kritikakou*, *Member, IEEE*

***Abstract*—Wireless Sensor and Actuator Networks (WSANs) are emerging as a new generation of Wireless Sensor Networks (WSNs). Due to the coupling between the sensing areas of the sensors and the action areas of the actuators, the efficient coordination among the nodes is a great challenge. In this paper, we address the problem of distributed node coordination in WSANs aiming at meeting the user's requirements on the states of the Points of Interest (POIs) in a real-time and energy-efficient manner. The node coordination problem is formulated as a non-linear program. To solve it efficiently, the problem is divided into two correlated subproblems: the Sensor-Actuator (S-A) coordination and the Actuator-Actuator (A-A) coordination. In the S-A coordination, a distributed federated Kalman filter-based estimation approach is applied for the actuators to collaborate with their ambient sensors to estimate the states of the POIs. In the A-A coordination, a distributed Lagrange-based control method is designed for the actuators to optimally adjust their outputs, based on the estimated results from the S-A coordination. The convergence of the proposed method is proved rigorously. As the proposed node coordination scheme is distributed, we find the optimal solution while avoiding high computational complexity. The simulation results also show that the proposed distributed approach is an efficient and practically applicable method with reasonable complexity.**

***Keywords*—Wireless sensor and actuator networks, distributed coordination, energy, delay, convergence analysis.**

I. Introduction

WITHIN the ten last years we have observed the remarkable growth of Wireless Sensor Networks (WSNs) in a variety of areas, such as environment monitoring, target tracking, object and event detection [2], [3]. The WSNs can be viewed as open-loop systems with low-cost wireless sensors to monitor their surrounding physical world. WSNs extended with wireless communication capable actuators become Wireless Sensor and Actuator Networks (WSANs). They can not only measure specific physical variables (e.g., temperature, illumination intensity, voltage), but also change them through a set of actions performed by the actuators. This is one of the key elements of the Internet of Things (IoT) [4], [5]. The sensors usually have limited power, sensing, computation and communication capabilities, while the actuators have higher capabilities as they require to make decisions and perform actions. The WSANs have found promising applications in environment control, building automation, industrial control and smart grid [6]–[13].

An important challenge in the design of such systems is that the sensing areas of the sensors and the action areas of the actuators are coupled, which introduces a significant difficulty in the efficient coordination of the sensors and the actuators [6], [7], [10]. A typical WSAN example is the industrial environment control [7], [10], where a number of wireless thermometers (sensors) and heaters/coolers (actuators) are deployed in a Region of Interest (ROI) to monitor and control the states (temperatures) of the Points of Interest (POIs). As the example shows in Fig. 1, the aim is to control the states of POIs P_1, P_2 and P_3 to meet the user's requirements (e.g., 23 °C). Therefore, the actuators A_1 and A_2 need to coordinate with each other using the information received from the sensors S_1, S_2 and S_3. Since the actuator A_3 and the POI P_3 are uncorrelated to the other actuators and POIs, the actuator A_3 is able to make control decision individually, using the information from the sensors S_4 and S_5. Hence, all the sensors and the actuators, whose sensing and action ranges that cover the same POIs, should coordinate with each other to decide for the appropriate actions.

A centralized scheme [14], [15] addresses the node coordination problem by gathering the information from all sensors and actuators and making the optimal state estimation and the optimal actuator control decision. However, the communication and the computation overhead is very high, especially when the network size is large. In addition, the risk of single node failure exists (e.g., at the sink or at the node near to the sink). As an alternative, a distributed node coordination is able to reduce the computational complexity and improve the system's scalability [7]. Distributed approaches are required at each phase of the WSAN coordination. Through a distributed Sensor-Actuator (S-A) coordination, the actuators collaborate with their ambient sensors to estimate the states of POIs (e.g., in Fig. 1, the actuators A_1 and A_2 collaborate with the sensors S_1, S_2 and S_3 to estimate the states of POIs P_1 and P_2). A distributed Actuator-Actuator (A-A) coordination allows the correlated actuators to coordinate with each other in order to adjust their outputs and control the states of POIs to meet the user's requirements (e.g., in Fig. 1, the actuators A_1 and A_2 coordinate with each other to control the states of POIs P_1 and P_2). Note that a closed-loop control is formed by the distributed S-A and A-A coordinations, which implies that they should be solved concurrently.

In WSANs, the number of sensors is usually much larger than the number of actuators. An actuator may receive information from multiple sensors [16]–[18]. Hence, a key issue for the distributed S-A coordination is to fuse the sensors information at the actuators, in order to estimate the real environment states

*INRIA/IRISA, Université de Rennes 1, 35042 Rennes, France. E-mail: lei.mo@inria.fr, angeliki.kritikakou@irisa.fr.

†School of Automation, Southeast University, 210096 Nanjing, China. E-mail: xhcao@seu.edu.cn. *(Correspondence Author)*

‡Université de Lorraine, CNRS, LORIA, 54000 Nancy, France. E-mail: yeqiong.song@loria.fr.

An earlier version of this paper was published in the IFAC World Congress [1].

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/JIOT.2018.2839030, IEEE Internet of Things Journal

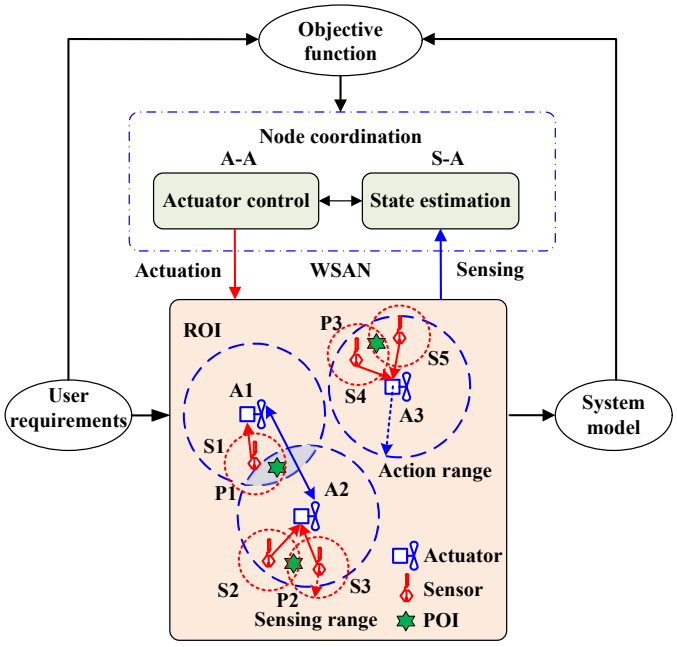


Fig. 1. Node coordination in WSANs.

(e.g., in Fig. 1, to estimate the state of POI P_3, the actuator A_3 should fuse the information from the sensors S_4 and S_5 rather than the information from the sensors S_1, S_2, or S_3). Although the distributed state estimation and the data fusion have been well studied in WSNs, e.g., [19], [20], these methods are hard to be directly applied to WSANs, since the locations and the outputs of actuators influence the state estimation. Previous works target at WSANs usually consider centralized schemes, e.g., [8], [16], [18], while distributed schemes are rare, e.g., [17], [21], and they focus on different contexts (see Section II for more details).

During the A-A coordination design, several important requirements should be taken into account. Although the actuators have more energy supply than the sensors, their energy consumption is still vital for a long-term control [14], [22]. In fact, sensors and actuators consume energy in different ways: the energy consumption of a sensor is mainly the wireless communications; while for an actuator, the energy is mainly used for performing the actions [23]. Hence, enhancing the energy efficiency of the actuators is an important issue. Besides the control accuracy requirements (i.e., the states of POIs should meet the user's requirements) [6], [7], [10], [11], [15], the actuator control should also satisfy the required real-time constraints. For example, if an abnormal temperature change occurs at some POIs, a swift action is required within a given time to re-stabilize the normal temperature. Hence, the core of the A-A coordination is to control the states of POIs to meet the user's requirements in a real-time and energy-efficient manner.

The linear A-A coordination problems, such as jointly optimize accuracy-delay [11] and accuracy-energy [15], can be optimally solved by Linear Matrix Inequality (LMI) [11] or Linear Quadratic Regulator (LQG) [15]. However, when multiple objectives are taken into account (e.g., control accuracy, energy efficiency and real-time performance), the A-A coordination problems are usually formulated as non-linear programmings. To solve these problems, the common methods include heuristics [14], [22], [24] and simulated annealing [25]. Although heuristics can find feasible solutions in a short amount of time, they do not provide bounds on the solution quality. In addition, they are not always easily extensible and when new assumptions or constraints must taken into account they must be redeveloped. Moreover, the methods in [14], [22], [25] are hard to be applied for a distributed control.

For WSANs deployed in harsh and critical environments, multiple requirements are implied (e.g., control accuracy, energy efficiency and real-time performance). Hence, the S-A and A-A coordination joint-design problem under such constraints is essential for WSANs. However, this joint consideration is rare in the existing literature, especially for the distributed solution. Hence, this paper answers the following questions: 1) How to find a proper way to formulate the S-A and A-A coordination joint-design problem so as to minimize the control error subject to energy supply and real-time constraints? and 2) Is there a way to achieve an optimal solution while avoiding high computational complexity? This paper is initially based on our preliminary results in [1] and significantly extends this work by providing the following contributions:

1) We formulate the WSAN node coordination problem as a non-linear problem, taking control accuracy, energy efficiency and real-time performance into account. The proposed system model covers the general case, compared with the simplified model of [1] which considers only diagonal matrices. The problem is proved to be convex and it is divided into two correlated S-A and A-A coordination subproblems, which can be solved sequentially.
2) Due to the general system model, the system states are now correlated with each other. Therefore, the S-A coordination scheme has to be modified in order to deal with the coupled system states compared to the approach presented in [1], where no state correlation is possible. A distributed S-A coordination solution is proposed using a Voronoi cells method to group the sensors and the actuators into several clusters. In each cluster, one actuator is selected as fusion center (cluster head), which is responsible for collaborating with its ambient sensors (cluster members) to perform state estimation. The received information from the sensors contains measurement noises, and, thus, the cluster head performs a distributed Federated Kalman Filter (FKF)-based method to optimally estimate the system states.
3) We propose a distributed Lagrange-based algorithm to solve the A-A coordination problem. This method coordinates the actuators that cover correlated POIs. Each actuator individually makes control decision, based on the limited exchange of information between the actuators. Moreover, the algorithm convergence is analyzed theoretically. The A-A coordination scheme presented in [1] is centralized, which prohibits its application in large size networks.
4) The theoretical analysis and the simulations show that the proposed method achieves the desirable control accuracy and real-time performance and energy efficiency. In this paper, we conduct extensive simulations to validate the robustness and the scalability of the proposed method under in non-idea environments (i.e., with faulty sensor measurements and actuator failures), compared with the simulations in [1] which only evaluate the system control performance. Moreover, the proposed method achieves an optimal state estimation and an optimal control scheme with low computation overhead

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/JIOT.2018.2839030, IEEE Internet of Things Journal

due to the distributed architecture.

The rest of this paper is organized as follows. Section II reviews the literature. Section III presents the system model and formulates the problem. Section IV and Section V describe the distributed S-A and A-A coordination schemes, respectively. Finally, Section VI shows the simulation results and Section VII concludes this paper.

II. Related Work

For centralized S-A coordination, the work in [16] proposes a fault-tolerant data routing and fusion method with minimum energy consumption under a delay bound. In [26], an event data collection approach is presented to balance network lifetime and event detection reliability. To avoid obstacles in the sensing field and collect data from the sensors, an energy-efficient routing mechanism is designed in [27] for the mobile data collectors. However, the state estimation is not considered in these works. When sensor readings are noisy, the quality of fusion result degrades. A Maximum Likelihood Estimation (MLE)-based method is proposed in [8] to locate the fire source, estimate the temperature and evaluate its emergency. A enhanced Bayesian inference based method is designed in [18] to fuse the sensory data and estimate the system states.

For distributed state estimation, some methods have been successfully applied to the WSNs, e.g., [19], [20]. Although these works also utilize Kalman filter to perform state estimation, they still differ from our method in the following ways. The estimation method in [19] focuses on smoothing the system states by a diffusion Kalman filter (i.e., estimating the states of discrete system at non-sampling steps). The scenario is different from ours since we consider the state estimation at sampling steps. In [20] all the sensors apply a consensus method to obtain the same state estimation in steady-state. Our method is not based on consensus, since in our scenario different sensors may be responsible for different system states. Moreover, the above methods are not suitable for WSANs since the closed-loop control provided by the actuators is not taken into account. In WSANs, although distributed data routing and fusion methods can be found in [17], [21], the state estimation is not considered in these works.

For centralized A-A coordination, a simplex based two-stage optimization method is proposed in [6] and an LQG based control algorithm is designed in [15] to jointly optimize the control accuracy and energy-efficiency. A simulated annealing based online approach is presented in [25] which jointly addresses the actuator control and communication scheduling problems. To complete the tasks with minimum energy consumption and under a given time delay, a heuristic based control scheme is proposed in [22] to adjust the actuators' outputs. A two-stage control scheme which is based on queuing theory is designed in [28] to enhance the energy efficiency and reduce the task completion time. A fault tolerant algorithm is provided in [29] to jointly optimize the scheduling, routing and control. A neural network based control algorithm is designed in [30] to achieve a precise light control and maintain a robustness against the inaccurate system parameters.

For distributed A-A coordination, the problems of controlling the indoor temperature to meet the user's requirements (control accuracy) are studied in [7], [10]. To balance the control accuracy and the energy consumption of an indoor lighting system, a gradient descent based control algorithm is proposed in [31]. A heuristic based control scheme is developed in [32] to minimize the workloads (energy consumption) of the actuators, while keeping them approximately balanced. To jointly optimize the accuracy-delay, an LMI-based control scheme [11] is applied to smart grid to implement a distributed voltage control. A heuristic based control scheme is presented in [24] to guarantee the desired levels of the control performance and reduce the energy consumption. However, the problems of control accuracy, energy-efficiency and real-timeliness are not considered simultaneously in above works.

III. System Model and Problem Formulation

In this section, we first present the WSAN model and then formulate the node coordination problem. A summary of the notations that are used in this paper is given as follows. For a matrix $\boldsymbol{M}$, m_{ij} is the $(i,j)^{th}$ element of $\boldsymbol{M}$, and $tr(\boldsymbol{M})$ is the trace of $\boldsymbol{M}$. For a vector $\boldsymbol{v}$, v_i is the i^{th} element of $\boldsymbol{v}$, and $\mathcal{D}(\boldsymbol{v})$ is the dimension of $\boldsymbol{v}$. For a scalar ϱ, $[\varrho]^+ = \max\{0, \varrho\}$, and $[\varrho]_{\underline{\varrho}}^{\overline{\varrho}} = \max\{\underline{\varrho}, \min\{\varrho, \overline{\varrho}\}\}$. $\mathbb{E}(\cdot)$, $(\cdot)'$, and $\|\cdot\|_2$ are the operators for the expectation of a random variable, the transpose of a matrix/vector, and the 2-norm of a vector, respectively. $\boldsymbol{y} \sim \mathcal{N}(\boldsymbol{\mu}, \boldsymbol{\sigma})$ represents the random variables $\boldsymbol{y}$ following a Gaussian distribution with mean $\boldsymbol{\mu}$ and covariance $\boldsymbol{\sigma}$, where $\boldsymbol{\mu}$ and $\boldsymbol{\sigma}$ have the appropriate dimensions. Let $\boldsymbol{p} = [p_1 \ldots, p_m]'$ and $\boldsymbol{q} = [q_1 \ldots, q_m]'$. $\boldsymbol{p} \preceq \boldsymbol{q}$ represents $p_i \leq q_i$ $(1 \leq i \leq m)$.

A. System Model

We consider n_s static sensors $\{S_1, \ldots, S_{n_s}\}$ and n_a static actuators $\{A_1, \ldots, A_{n_a}\}$ randomly deployed in an ROI to monitor the states of n_p POIs and take necessary actions to deal with the events occurring in these POIs. The system states $\boldsymbol{x} = [x_1, \ldots, x_{n_p}]'$ represent the physical variables under consideration (e.g., temperature, illumination intensity, voltage). An event is occurred when a system state exceeds a predefined threshold. The sensors measure the system states periodically and transmit their readings to the ambient actuators. Based on the collected information, the actuators coordinate with each other to make proper decisions and perform the corresponding actions to control the system states to meet the user's requirements $\boldsymbol{x}^*$.

Note that 1) a system state x_i is affected by both the nearby actuators and the correlated system states, 2) the actuators are static, and 3) the information exchanging between the sensors and the actuators is carried out by discrete wireless packets [7], [10]. We consider a system which is modeled by the following discrete Linear Time-Invariant (LTI) process:

$$\boldsymbol{x}(k+1) = \boldsymbol{A}\boldsymbol{x}(k) + \boldsymbol{B}\boldsymbol{u}(k) + \boldsymbol{\omega}(k), \tag{1}$$

where $\boldsymbol{u}(k) \in R^{n_a \times 1}$ and $\boldsymbol{\omega}(k) \sim \mathcal{N}(\boldsymbol{0}, \boldsymbol{Q})$ are the vectors of actuators' outputs and system noise, respectively. $\boldsymbol{A} \in R^{n_p \times n_p}$ and $\boldsymbol{B} \in R^{n_p \times n_a}$ are the system matrix and the input matrix, respectively. Δ is the system sampling period, and $\underline{\boldsymbol{u}} \preceq \boldsymbol{u}(k) \preceq \overline{\boldsymbol{u}}$ since the actuators' outputs are bounded.

For the sensor S_j, if the system states $\{x_p, \ldots, x_q\}$ are within its sensing range r_s, the sensor S_j takes a noisy measurement of them:

$$\begin{aligned}\boldsymbol{z}_j(k) =& [c_p^j x_p(k) + \nu_p^j(k), \ldots, c_q^j x_q(k) + \nu_q^j(k)]' \\ =& [z_p^j(k), \ldots, z_q^j(k)]',\ 1 \leq j \leq n_s,\ 1 \leq p, q \leq n_p,\end{aligned} \tag{2}$$

where c_i^j $(p \leq i \leq q)$ is a coefficient, ν_i^j is the measurement noise, and $\boldsymbol{\nu}(k) \sim \mathcal{N}(\boldsymbol{0}, \boldsymbol{R})$. We assume that the system

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/JIOT.2018.2839030, IEEE Internet of Things Journal

described by (1) and (2) is controllable and observable and it is synchronized by one of the existing clock synchronization methods [33].

B. Problem Formulation

Our objective is to find, for each occurring event, the strategies of the sensors and the actuators to drive the system states $\boldsymbol{x}$ towards the user's requirements $\boldsymbol{x}^*$ subject to a set of real-time and energy supply constraints. The whole problem is divided into two correlated subproblems of the S-A and the A-A coordinations described as follows.

Denote $\boldsymbol{z}(k) = [\boldsymbol{z}_1(k)', \ldots, \boldsymbol{z}_{n_s}(k)']'$ as the collected sensor measurements of the entire network at the k^{th} step. The goal of the S-A coordination is to determine a posteriori estimate $\boldsymbol{x}(k|k)$ and a priori estimate $\boldsymbol{x}(k|k-1)$ of system states $\boldsymbol{x}(k)$ [34] as defined below:

$$\boldsymbol{x}(k|k) = \mathbb{E}(\boldsymbol{x}(k)|\boldsymbol{z}(k)),$$
$$\boldsymbol{x}(k|k-1) = \mathbb{E}(\boldsymbol{x}(k)|\boldsymbol{z}(k-1)).$$

Definition 3.1 (Event): Denote ϵ_i as a small positive tolerance. If the state estimation $x_i(k|k)$ exceeds a predefined range $[x_i^* - \epsilon_i, x_i^* + \epsilon_i]$, while the state estimation $x_i(k-1|k-1)$ was within this range, an event in the system state x_i is occurred at the k^{th} step.

Definition 3.2 (Actuation delay): For an event in the system state x_i, the actuation delay is the number of sampling periods that the state estimation $x_i(k|k)$ remains out of the range $[x_i^* - \epsilon_i, x_i^* + \epsilon_i]$ after the occurrence of this event.

Assume that an event occurs in system state x_i at the k^{th} step. Let $T_i(k)$ be the actuation delay, $\mathcal{A}_i = \{A_j | b_{ij} \neq 0, 1 \leq j \leq n_a\}$ be the set of responsible actuators of system state x_i, and $\mathcal{U}_i(k) = \{u_j(k) | b_{ij} \neq 0, 1 \leq j \leq n_a\}$ be the set of corresponding actuators' outputs. Note that 1) $T_i(k)$ mainly depends on the control error $e_i(k) = x^* - x_i(k|k)$ and the actuators' outputs $\mathcal{U}_i(k)$, and 2) under the given control error $e_i(k)$, if the actuator's output $u_j(k)$ ($b_{ij} \neq 0$) increases, the actuation delay $T_i(k)$ reduces. Hence, we have

$$T_i(k) = \mathcal{G}(e_i(k), \mathcal{U}_i(k)),$$

where $\mathcal{G}$ is a differentiable and decreasing function with respect to $\mathcal{U}_i$. We assume that the elements in the set $\mathcal{U}_i$ affect the value of T_i independently (i.e., $\frac{\partial \mathcal{G}^2}{\partial u_p \partial u_q} = 0$ $(p \neq q)$).

To drive the system states $\boldsymbol{x}$ towards the user's requirements $\boldsymbol{x}^*$, the actuators' outputs $\boldsymbol{u}(k)$ can be either positive or negative. Denote $p_j(k) = k_p u_j(k)^2$ as the actuator A_j's power at the k^{th} step, where k_p is a positive constant (i.e., the larger actuator output is, the more energy is consumed). Since each actuator maintains its output step-wise, the consumed energy of the actuator A_j at the k^{th} step is given by $E_j(k) = \Delta p_j(k)$.

Our objective is to minimize the control error $\|\boldsymbol{A}\boldsymbol{x}(k|k) + \boldsymbol{B}\boldsymbol{u}(k) - \boldsymbol{x}^*\|_2^2$ and the energy consumption of the actuators $\sum_{j=1}^{n_a} \Delta k_p u_j(k)^2$ by adjusting the actuators' outputs $\boldsymbol{u}(k)$. Summarizing the objective and all the constraints mentioned above, the Primal Problem (PP) is formulated as

$$\textbf{PP}: \min_{\boldsymbol{u}(k)} \ \mathcal{J}(k) \tag{3}$$
$$\text{s.t.} \begin{cases} \Delta k_p u_j(k)^2 \leq E_j^r(k), \ 1 \leq j \leq n_a, \\ G(e_i(k), \mathcal{U}_i(k)) + T_d \leq T_{th}, \ 1 \leq i \leq n_p, \\ \underline{u}_j \leq u_j(k) \leq \overline{u}_j, \ 1 \leq j \leq n_a, \end{cases}$$

where

$$\mathcal{J}(k) = \|\boldsymbol{A}\boldsymbol{x}(k|k) + \boldsymbol{B}\boldsymbol{u}(k) - \boldsymbol{x}^*\|_2^2 + \alpha \sum_{j=1}^{n_a} \Delta k_p u_j(k)^2,$$

and α is a positive coefficient. $\Delta k_p u_j(k)^2 \leq E_j^r(k)$ indicates that the energy consumption of each actuator cannot exceed its residual energy $E_j^r(k)$. $\mathcal{G}(e_i(k), \mathcal{U}_i(k)) + T_d \leq T_{th}$ implies that the actuation delay should not exceed a time threshold T_{th}, minus by the communication and computing time T_d. T_{th} is determined by the applications and $T_{th} \geq \Delta$.

Theorem 3.1: The PP (3) is a convex optimization problem.

Proof: See Appendix A for the proof. ■

The PP requires an appropriate A-A coordination to solve the actuator output adjustment problem based on the state estimation $\boldsymbol{x}(k|k)$, which is obtained by the S-A coordination. In the following section, we will show the details of S-A coordination and A-A coordination.

IV. S-A Coordination and Distributed Estimation

In this section, a distributed fusion method is applied among the actuators and their ambient sensors to estimate the system states, based on the FKF [35]. As shown in Fig. 2, this method employs a two-stage data processing structure, where the Local Filters (LFs) (i.e., $\{\text{LF}_1, \ldots, \text{LF}_h\}$) perform the local state estimation using the measurements received from the sensors. Then, the estimated results from the LFs are fused in a Master Filter (MF) to yield an optimal state estimation.

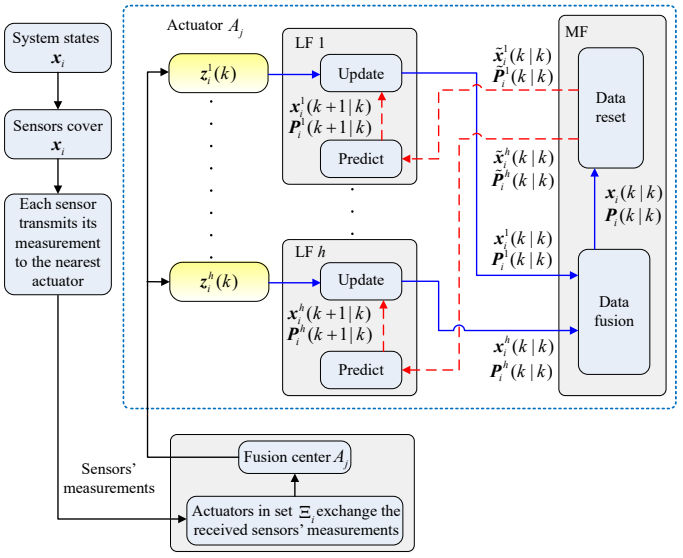


Fig. 2. The structure of FKF design for S-A coordination.

A. Preliminaries

Denote r_x as the impact range of the system state x_i, and r_a as the action range of the actuator A_j. Usually, the values of r_x and r_a are limited and much smaller than the range of ROI. Hence, if the POIs and the actuators are randomly deployed in the ROI rather than concentrated on a small area, the control of the system states x_p and x_q $(p \neq q)$ may be decoupled with each other (e.g., in Fig. 1, the control of system states $\{x_1, x_2\}$ and x_3 are decoupled). Let $\mathcal{X}_i = \{x_j | a_{ij} \neq 0, 1 \leq j \leq n_p, j \neq i\}$ be the set of correlated system states of x_i. Based on the elements in the sets $\{\mathcal{X}_i, \mathcal{A}_i\}$ $(1 \leq i \leq n_p)$, the system states $\boldsymbol{x}$

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/JIOT.2018.2839030, IEEE Internet of Things Journal

can be rearranged as $[\boldsymbol{x}_1', \ldots, \boldsymbol{x}_\kappa']'$ $(1 \leq \kappa \leq n_p)$. The elements in $\boldsymbol{x}_i$ have the following properties: for x_i^p and x_j^q $(\forall p, \forall q)$, if $i \neq j$, we have $\mathcal{X}_p \cap \mathcal{X}_q = \emptyset$ and $\mathcal{A}_p \cap \mathcal{A}_q = \emptyset$, where x_i^j represents the j^{th} element of $\boldsymbol{x}_i$. Due to the independence between the system states $\boldsymbol{x}_i$ and $\boldsymbol{x}_j$, we focus on the node coordination design for $\boldsymbol{x}_i$. To avoid confusion, we redefine the subscripts of the elements in $\boldsymbol{x}_i$ as $\boldsymbol{x}_i = [x_1, \ldots, x_{\mathcal{D}(\boldsymbol{x}_i)}]'$. Hence, the dynamic model of system states $\boldsymbol{x}_i$ is given by

$$\boldsymbol{x}_i(k+1) = \boldsymbol{A}_i \boldsymbol{x}_i(k) + \boldsymbol{B}_i \boldsymbol{u}_i(k) + \boldsymbol{\omega}_i(k), \ 1 \leq i \leq \kappa, \quad (4)$$

where $\boldsymbol{A}_i$, $\boldsymbol{B}_i$, $\boldsymbol{u}_i$ and $\boldsymbol{\omega}_i(k)$ are the corresponding parts of $\boldsymbol{A}$, $\boldsymbol{B}$, $\boldsymbol{u}$ and $\boldsymbol{\omega}(k)$, respectively, under the given $\boldsymbol{x}_i$, and $\boldsymbol{\omega}_i(k) \sim \mathcal{N}(\boldsymbol{0}, \boldsymbol{Q}_i)$.

Denote $\boldsymbol{\mathcal{A}}_i$ as the set of responsible actuators of the system states $\boldsymbol{x}_i$. If $\boldsymbol{x}^* - \boldsymbol{\epsilon} \leq \boldsymbol{x}_i \leq \boldsymbol{x}^* + \boldsymbol{\epsilon}$, we have $\boldsymbol{u}_i = 0$, else the actuators in the set $\boldsymbol{\mathcal{A}}_i$ should coordinate with each other to perform appropriate actions, since the control of system states x_i^p and x_i^q $(p \neq q)$ are now correlated. Without loss of generality, we assume that at least one event occurs in the system states $\boldsymbol{x}_i$ at the k^{th} step.

Using the Voronoi cells method [32], the ROI can be divided into n_a partitions, where all the sensors in the j^{th} partition are closer to the actuator A_j than to all other actuators. Hence, the actuator A_j periodically collects the measurements from the sensors in its own partition. This is because: 1) the nearest actuator is probably the one that covers the corresponding POI of the reported event, and, thus, it can take the action to react to this event quickly, and 2) the transmission over the shortest path is expected to suffer less interference and low packet loss, and, thus, it has a higher reliability. As the example shows in Fig. 3, the actuators A_1 and A_2 are the nearest actuators to the POIs P_1 and P_2, as well as the responsible actuators of the system states x_1 and x_2.

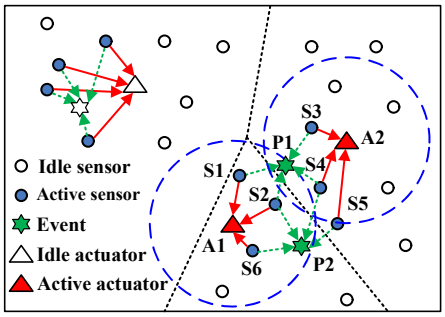


Fig. 3. Distributed S-A coordination illustration.

Since the sensors that cover the system states $\boldsymbol{x}_i$ may belong to different partitions, multiple actuators can receive the measurements of the system states $\boldsymbol{x}_i$. We group these actuators as a set (cluster) Ξ_i, and select one actuator in the set Ξ_i as the fusion center (cluster head). For example, in Fig. 3, the actuator A_1 receives the measurements of the system states x_1 and x_2 from the sensors S_1, S_2 and S_6, while the actuator A_2 receives the measurements of the system states x_1 and x_2 from the sensors S_3, S_4 and S_5. Therefore, the actuators A_1 and A_2 are grouped as a cluster Ξ_1 and we select the actuator A_1 as the cluster head. The fusion center selection can be implemented in a fully distributed manner [36]. To perform the data fusion, the cluster members send their received measurements to the cluster head. Note that the elements in the sets Ξ_i and Ξ_j $(i \neq j)$ may be overlapping. To reduce the communication overhead of a single actuator, we consider one actuator cannot be selected as fusion center in different clusters (e.g., if the actuator A_1 is the fusion center in the cluster Ξ_1 and $A_1 \in \Xi_2$, the actuator A_1 cannot be selected as fusion center in the cluster Ξ_2).

B. FKF-based Data Fusion

Since the number of sensors is usually larger than the number of POIs (i.e., $n_s \gg n_p$), the system state x_p could be sensed by the multiple sensors (i.e., several measurements of system state x_p exist). Denote $\mathcal{Z}_p(k) = \{z_p^j(k) | c_p^j \neq 0, 1 \leq j \leq n_s\}$ as the set of measurements of system state x_p, and $\mathcal{Z}_p^j(k)$ as the j^{th} element of $\mathcal{Z}_p(k)$. Hence, $\boldsymbol{z}_i^j(k) = [\mathcal{Z}_1^j(k), \ldots, \mathcal{Z}_{\mathcal{D}(\boldsymbol{x}_i)}^j(k)]'$ represents the j^{th} measurement vector of the system states $\boldsymbol{x}_i$, which is given by

$$\boldsymbol{z}_i^j(k) = \boldsymbol{C}_i^j \boldsymbol{x}_i(k) + \boldsymbol{\nu}_i^j(k),$$

where $\boldsymbol{C}_i^j$ and $\boldsymbol{\nu}_i^j$ have compatible dimensions, and $\boldsymbol{\nu}_i^j(k) \sim \mathcal{N}(\boldsymbol{0}, \boldsymbol{R}_i^j)$. Since the sensors are randomly deployed, the number of elements in the set $\mathcal{Z}_i$ and in the set $\mathcal{Z}_j$ $(j \neq i)$ are not always the same. We assume that h independent measurement vectors of $\boldsymbol{x}_i$ can be obtained from the sets $\{\mathcal{Z}_1, \ldots, \mathcal{Z}_{\mathcal{D}(\boldsymbol{x}_i)}\}$. Hence, h LFs are implemented in parallel. As the example shows in Fig. 3, let $\boldsymbol{x}_i = [x_1, x_2]'$, since $\mathcal{Z}_1 = \{z_1^1, z_1^2, z_1^3, z_1^4\}$ and $\mathcal{Z}_2 = \{z_2^2, z_2^4, z_2^5, z_2^6\}$, we have $h = 4$. The fusion process is summarized as follows.

1) Based on the fusion results (S-A coordination) and the actuator outputs (A–A coordination) at the previous step $k-1$ (i.e., $\{\tilde{\boldsymbol{x}}_i^j(k-1|k-1), \tilde{\boldsymbol{P}}_i^j(k-1|k-1), \tilde{\boldsymbol{Q}}_i^j(k-1)\}$ and $\boldsymbol{u}_i(k-1)$), and the sensor measurements at the current step k (i.e. $\boldsymbol{z}_i^j(k)$), the j^{th} LF $(1 \leq j \leq h)$ performs the following state estimation:

$$\boldsymbol{P}_i^j(k|k-1) = \boldsymbol{A}_i \tilde{\boldsymbol{P}}_i^j(k-1|k-1) \boldsymbol{A}_i' + \tilde{\boldsymbol{Q}}_i^j(k-1), \quad (5)$$

$$\boldsymbol{x}_i^j(k|k-1) = \boldsymbol{A}_i \tilde{\boldsymbol{x}}_i^j(k-1|k-1) + \boldsymbol{B}_i \boldsymbol{u}_i(k-1), \quad (6)$$

$$\boldsymbol{P}_i^j(k|k)^{-1} = \boldsymbol{P}_i^j(k|k-1)^{-1} + \boldsymbol{C}_i^{j'} \boldsymbol{R}_i^{j^{-1}} \boldsymbol{C}_i^j, \quad (7)$$

$$\boldsymbol{x}_i^j(k|k) = \boldsymbol{P}_i^j(k|k)(\boldsymbol{P}_i^j(k|k-1)^{-1} \boldsymbol{x}_i^j(k|k-1) + \boldsymbol{C}_i^{j'} \boldsymbol{R}_i^{j^{-1}} \boldsymbol{z}_i^j(k)), \quad (8)$$

where $\boldsymbol{x}_i^j(k|k)$ and $\boldsymbol{x}_i^j(k+1|k)$ are the a posteriori estimate and the a priori estimate of the system states $\boldsymbol{x}_i(k)$ obtained by the j^{th} LF, respectively. $\boldsymbol{P}_i^j(k|k)$ and $\boldsymbol{P}_i^j(k+1|k)$ are the error covariance matrices corresponding to the state estimations $\boldsymbol{x}_i^j(k|k)$ and $\boldsymbol{x}_i^j(k+1|k)$, respectively.

2) Collecting the estimation results $\{\boldsymbol{x}_i^j(k|k), \boldsymbol{P}_i^j(k|k)\}$ $(1 \leq j \leq h)$ from the LFs, the MF performs the following data fusion process:

$$\boldsymbol{P}_i(k|k)^{-1} = \sum\nolimits_{j=1}^{h} \boldsymbol{P}_i^j(k|k)^{-1}, \quad (9)$$

$$\boldsymbol{x}_i(k|k) = \boldsymbol{P}_i(k|k) \sum\nolimits_{j=1}^{h} (\boldsymbol{P}_i^j(k|k)^{-1} \boldsymbol{x}_i^j(k|k)). \quad (10)$$

3) The MF resets the fusion results $\boldsymbol{x}_i(k|k)$ and $\boldsymbol{P}_i(k|k)$ as follows:

$$\tilde{\boldsymbol{x}}_i^j(k|k) = \boldsymbol{x}_i(k|k), \quad (11)$$

$$\tilde{\boldsymbol{P}}_i^j(k|k) = \beta_i^j(k)^{-1} \boldsymbol{P}_i(k|k), \quad (12)$$

$$\tilde{\boldsymbol{Q}}_i^j(k) = \beta_i^j(k)^{-1} \boldsymbol{Q}_i, \quad (13)$$

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/JIOT.2018.2839030, IEEE Internet of Things Journal

where $\beta_i^j(k)$ is called the information-sharing factor and it must satisfies the constraints $0 \leq \beta_i^j(k) \leq 1$ and $\sum_{j=1}^{h} \beta_i^j(k) = 1$ to avoid information loss. We select the value of $\beta_i^j(k)$ based on the estimation accuracy of the j^{th} LF [1] (e.g., $\beta_i^j(k) = \frac{(tr(\boldsymbol{P}_i^j(k|k)))^{-1}}{\sum_{j=1}^{h}(tr(\boldsymbol{P}_i^j(k|k)))^{-1}}$).

4) The MF sends the reseted results $\tilde{\boldsymbol{x}}_i^j(k|k)$, $\tilde{\boldsymbol{P}}_i^j(k|k)$ and $\tilde{\boldsymbol{Q}}_i^j(k)$ back to the j^{th} LF for the next round of data fusion.

Lemma 4.1: The fusion result (10) provided by the MF is an optimal estimation of the system states $\boldsymbol{x}_i(k)$.

Proof: The proof is similar to that in [37], where the FKF with data reset (11) – (13) is equal to the centralized Kalman filter. ■

(4) and (10) show that 1) κ fusion centers perform state estimation simultaneously, and 2) each fusion center is only responsible for the estimation of partial system states (e.g., $\boldsymbol{x}_i$). Hence, the proposed S-A coordination is distributed. The implementation details are summarized in Algorithm 1. Note that 1) the actuators' outputs are not known at the sensors, and 2) the actuators are more powerful than the sensors. In order to reduce the amount of message exchanges among the sensors and the actuators and make a better usage of nodes' resources, the computationally expensive tasks, such as state estimation and data fusion, are performed by the actuators rather than the sensors.

Algorithm 1: Distributed S-A coordination

```
Set initial values: x(0) and P(0);
for k = 1, 2, 3, ... do
    for i = 1, ..., κ do
        for Actuators in set A_i do
            Receive the data transmitted from the
              sensors in its own partition;
            if Actuator A_l is fusion center then
                Receive the sensor data transmitted from
                  the other actuators in the cluster Ξ_i;
                Construct {z_i^1(k), ..., z_i^h(k)};
                for j = 1, ..., h do
                    Update {x_i^j(k|k), P_i^j(k|k)} through
                      (5) – (8) ;
                end
                Fuse {x_i^j(k|k), P_i^j(k|k)} (1 ≤ j ≤ h)
                  through (9) – (10);
                for j = 1, ..., h do
                    Reset {x_i(k|k), P_i(k|k)} through
                      (11) – (13);
                end
            else
                Relay the received sensor data to the
                  fusion center in the cluster Ξ_i;
            end
            Calculate u_l(k) through the A-A
              coordination;
        end
    end
end
```

V. A-A COORDINATION AND DISTRIBUTED CONTROL

In this section, we describe the distributed algorithm for solving the PP (3). Our method is based on dual decomposition and we solve the dual problem using the subgradient algorithm.

To develop the dual problem of PP, we introduce the positive Lagrange multipliers $\boldsymbol{\lambda} = [\lambda_1, \ldots, \lambda_{n_a+n_p}]'$ to its constraints, the corresponding Lagrangian [38] is

$$\begin{aligned}\mathcal{L}(\boldsymbol{u}(k), \boldsymbol{\lambda}) =& \|\boldsymbol{A}\boldsymbol{x}(k|k) + \boldsymbol{B}\boldsymbol{u}(k) - \boldsymbol{x}^*\|_2^2 \\ &+ \alpha \sum\nolimits_{j=1}^{n_a} E_j(k) + \sum\nolimits_{j=1}^{n_a} \lambda_j (E_j(k) - E_j^r) \\ &+ \sum\nolimits_{j=n_a+1}^{n_a+n_p} \lambda_j (T_j(k) + T_d - T_{th}) \\ =& \sum\nolimits_{i=1}^{\kappa} \mathcal{L}_i(\boldsymbol{u}_i(k), \boldsymbol{\lambda}_i), \end{aligned} \tag{14}$$

where $\boldsymbol{\lambda}_i = [\lambda_1, \ldots, \lambda_{\mathcal{D}(\boldsymbol{u}_i)}, \lambda_{\mathcal{D}(\boldsymbol{u}_i)+1}, \ldots, \lambda_{\mathcal{D}(\boldsymbol{\lambda}_i)}]'$ and

$$\begin{aligned}\mathcal{L}_i(\boldsymbol{u}_i(k), \boldsymbol{\lambda}_i) =& \|\boldsymbol{A}_i\boldsymbol{x}_i(k|k) + \boldsymbol{B}_i\boldsymbol{u}_i(k) - \boldsymbol{x}_i^*\|_2^2 \\ &+ \alpha \sum\nolimits_{j=1}^{\mathcal{D}(\boldsymbol{u}_i)} E_j(k) + \sum\nolimits_{j=1}^{\mathcal{D}(\boldsymbol{u}_i)} \lambda_j (E_j(k) - E_j^r) \\ &+ \sum\nolimits_{j=\mathcal{D}(\boldsymbol{u}_i)+1}^{\mathcal{D}(\boldsymbol{\lambda}_i)} \lambda_j (T_j(k) + T_d - T_{th}). \end{aligned}$$

In (14), $\boldsymbol{x}(k|k) = [\boldsymbol{x}_1(k|k)', \ldots, \boldsymbol{x}_\kappa(k|k)']'$ and $\boldsymbol{x}_i(k|k)$ $(1 \leq i \leq \kappa)$ is given by (10).

The dual function $\mathcal{R}(\boldsymbol{\lambda})$ is defined as the minimum value of the Lagrangian $\mathcal{L}(\boldsymbol{u}(k), \boldsymbol{\lambda})$ over the variables $\boldsymbol{u}$, i.e., for $\boldsymbol{\lambda}$,

$$\begin{aligned}\mathcal{R}(\boldsymbol{\lambda}) &= \min_{\underline{\boldsymbol{u}} \preceq \boldsymbol{u}(k) \preceq \overline{\boldsymbol{u}}} \mathcal{L}(\boldsymbol{u}(k), \boldsymbol{\lambda}) = \sum\nolimits_{i=1}^{\kappa} \mathcal{R}_i(\boldsymbol{\lambda}_i) \\ &= \sum\nolimits_{i=1}^{\kappa} \min_{\underline{\boldsymbol{u}}_i \preceq \boldsymbol{u}_i(k) \preceq \overline{\boldsymbol{u}}_i} \mathcal{L}_i(\boldsymbol{u}_i(k), \boldsymbol{\lambda}_i). \end{aligned} \tag{15}$$

Hence, the Dual Problem (DP) associated with the PP is

$$\textbf{DP}: \max_{\boldsymbol{\lambda} \succeq 0} \quad \mathcal{R}(\boldsymbol{\lambda}).$$

Since the PP is convex, the optimal objective function values of the PP and its dual problem are equivalent due to the strong duality (i.e., solving the DP is equal to solving the PP). Based on the structure of the DP, we design a two-layer subgradient-based algorithm to solve this problem. The inner-layer iteration aims to update variables $\boldsymbol{u}$ under the given multipliers $\boldsymbol{\lambda}$. On the other hand, the outer-layer iteration aims to update multipliers $\boldsymbol{\lambda}$ under the given variable $\boldsymbol{u}$. Therefore, we introduce the indexes m and l to count the outer-layer and the inner-layer iterations, respectively.

1) Inner-layer Iteration: Assume that the current outer-layer iteration is m. Under the given multipliers $\boldsymbol{\lambda}(m)$, the actuator's output u_j (the j^{th} element in $\boldsymbol{u}_i$) is iteratively updated by

$$\begin{aligned}& u_j(l+1) \\ &= \left[u_j(l) - \delta \frac{\partial \mathcal{L}(\boldsymbol{u}(l), \boldsymbol{\lambda}(m))}{\partial u_j(l)} \right]_{\underline{u}_j}^{\overline{u}_j} \\ &= \left[u_j(l) - \delta \frac{\partial \mathcal{L}_i(\boldsymbol{u}_i(l), \boldsymbol{\lambda}_i(m))}{\partial u_j(l)} \right]_{\underline{u}_j}^{\overline{u}_j}, \ 1 \leq j \leq \mathcal{D}(\boldsymbol{u}_i), \end{aligned} \tag{16}$$

where δ is a positive small step-size, and

$$\begin{aligned}& \frac{\partial \mathcal{L}_i(\boldsymbol{u}_i(l), \boldsymbol{\lambda}_i(m))}{\partial u_j(l)} \\ =& 2 \sum\nolimits_{i=1}^{\mathcal{D}(\boldsymbol{x}_i)} \left(\sum\nolimits_{j=1}^{\mathcal{D}(\boldsymbol{x}_i)} a_{ij} x_j + \sum\nolimits_{j=1}^{\mathcal{D}(\boldsymbol{u}_i)} b_{ij} u_j(l) - x_i^* \right) b_{ij} \\ &+ 2(\alpha + \lambda_j(m)) \Delta k_p u_j(l) + \sum\nolimits_{j=\mathcal{D}(\boldsymbol{u}_i)+1}^{\mathcal{D}(\boldsymbol{\lambda}_i)} \lambda_j(m) \frac{\partial T_j(l)}{\partial u_j(l)}. \end{aligned}$$

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/JIOT.2018.2839030, IEEE Internet of Things Journal

Denote $\varepsilon \geq 0$ as a small tolerance. Since $\mathcal{L}(\boldsymbol{u}, \boldsymbol{\lambda})$ is a convex function with respect to $\boldsymbol{u}$, the inner-layer iteration can start from an arbitrary initial point $\boldsymbol{u}(0)$ and the iteration stops when $\|\boldsymbol{u}(l+1) - \boldsymbol{u}(l)\|_2^2 \leq \varepsilon$. Hence, by iterating (16), we obtain $\boldsymbol{u}(m)$ (the optimal solution of the problem (15) under the given multipliers $\boldsymbol{\lambda}(m)$).

2) Outer-layer Iteration: Based on the updated results of the previous inner-layer and outer-layer iterations (i.e., $\boldsymbol{u}(m)$ and $\boldsymbol{\lambda}(m)$), the multiplier λ_j (the j^{th} element in $\boldsymbol{\lambda}_i$) is iteratively updated by

$$\begin{aligned} &\lambda_j(m+1) \\ &= \left[\lambda_j(m) + \delta \frac{\partial \mathcal{R}\left(\boldsymbol{\lambda}(m)\right)}{\partial \lambda_j(m)}\right]^+ \\ &= \left[\lambda_j(m) + \delta \frac{\partial \mathcal{R}_i\left(\boldsymbol{\lambda}_i(m)\right)}{\partial \lambda_j(m)}\right]^+, \ 1 \leq j \leq \mathcal{D}(\boldsymbol{\lambda}_i), \end{aligned} \tag{17}$$

where

$$\frac{\partial \mathcal{R}_i(\boldsymbol{\lambda}_i(m))}{\partial \lambda_j(m)} = \begin{cases} E_j(m) - E_j^r, 1 \leq j \leq \mathcal{D}(\boldsymbol{u}_i), \\ T_j(m) + T_d - T_{th}, \mathcal{D}(\boldsymbol{u}_i) + 1 \leq j \leq \mathcal{D}(\boldsymbol{\lambda}_i). \end{cases}$$

Since $\mathcal{R}(\boldsymbol{\lambda})$ is a concave function with respect to $\boldsymbol{\lambda}$, the outer-layer iteration can start from an arbitrary initial point $\boldsymbol{\lambda}(0)$ and the iteration stops when $\|\boldsymbol{\lambda}_i(m+1) - \boldsymbol{\lambda}_i(m)\|_2^2 \leq \varepsilon$.

(16) show that $\frac{\partial \mathcal{L}(\boldsymbol{u}(l), \boldsymbol{\lambda}(m))}{\partial u_j(l)}$ is a function respective to variables $\boldsymbol{u}_i(l)$. Hence, during the inner-layer iteration, the actuator A_j needs to exchange its temporary result $u_j(l)$ with the other actuators in the set $\boldsymbol{\mathcal{A}}_i$ to update $u_j(l+1)$. On the other hand, (17) shows that $\frac{\partial \mathcal{R}(\boldsymbol{\lambda}(m))}{\partial \lambda_j(m)}$ is a function respective to variables $\boldsymbol{u}_i(m)$ and $\lambda_j(m)$. Note that $\boldsymbol{u}_i(m)$ is already known by the actuator A_j through the inner-layer iteration. The update of multipliers $\boldsymbol{\lambda}_i(m)$ can be performed locally by each actuator in the set $\boldsymbol{\mathcal{A}}_i$. (16) and (17) show that 1) the actuator A_j only needs to exchange information with the actuators in the set $\boldsymbol{\mathcal{A}}_i$ rather than all the actuators, and 2) the actuator A_j calculates its output individually based on the received information. Hence, the proposed A-A coordination is distributed. The implementation details are summarized in Algorithm 2.

Since the PP is convex, the optimal values of $\boldsymbol{\lambda}$ and $\boldsymbol{u}$ exist. Denote them as $\boldsymbol{\lambda}^*$ and $\boldsymbol{u}^*$, respectively. The convergence of proposed algorithm is analyzed through the following theorems.

Definition 5.1 (Statistical convergence [39]): For an optimization problem $\min_{\boldsymbol{\iota}} \mathcal{S}(\boldsymbol{\iota})$ and an iterative algorithm with positive step-size ς, which generates an intermediate solution $\boldsymbol{\iota}(m)$ at the m^{th} step, let $\boldsymbol{\iota}^*$ be the optimal solution and $\bar{\boldsymbol{\iota}}(m) = \frac{1}{m}\sum_{\tau=1}^{m} \boldsymbol{\iota}(\tau)$ be the average solution found by the m^{th} step. $\boldsymbol{\iota}$ is said to statistically converge to $\boldsymbol{\iota}^*$, if for any $\gamma > 0$ there exists an ς such that $\limsup_{m\to\infty}(\mathcal{S}(\bar{\boldsymbol{\iota}}(m)) - \mathcal{S}(\boldsymbol{\iota}^*)) \leq \gamma$.

Theorem 5.1: Iterating (17), the Lagrange multipliers $\boldsymbol{\lambda}$ statistically converge to $\boldsymbol{\lambda}^*$, when δ is a small enough value.

Proof: See Appendix B for the proof. ■

Theorem 5.2: Iterating (16) and (17), the actuators' outputs $\boldsymbol{u}$ statistically converge to $\boldsymbol{u}^*$, when δ is a small enough value.

Proof: See Appendix C for the proof. ■

From the above statements, we can see that there exist two types of information exchange among the actuators: 1) the actuators in the set Ξ_i send their received measurements to the fusion center for the data fusion, and 2) the actuators in the set $\boldsymbol{\mathcal{A}}_i$ exchange their temporary results $\boldsymbol{u}_i(l)$ with each other to calculate their outputs $\boldsymbol{u}_i(k)$. The former one only requires to exchange information once at each step, while the latter one requires several times at each step. Based on different positions of nodes and POIs, the actuators in the sets Ξ_i and $\boldsymbol{\mathcal{A}}_i$ are not always the same. According to the actuators in the sets $\{\Xi_i\}$ and $\{\boldsymbol{\mathcal{A}}_i\}$, each actuator knows which actuators should communicate with. Hence, the topology of communication between the actuators can be determined by the available methods [36]. Moreover, by Algorithm 2, each actuator in the set $\boldsymbol{\mathcal{A}}_i$ knows the values of $\boldsymbol{u}_i(k)$, which can be used for the next round data fusion.

Algorithm 2: Distributed A-A coordination

```
Set the initial values: u(0), λ(0), δ, and ε;
for k = 1, 2, 3, ... do
    for i = 1, ..., κ do
        for Actuator A_j, j = 1, ..., D(u_i) do
            m ← 1;
            Update λ_i(m) by (17);
            if ||λ_i(m) − λ_i(m−1)||_2^2 ≥ ε then
                Update λ_i(m+1) by (17);
                l ← 1;
                Update u_j(l) by (16);
                if ||u_i(l) − u_i(l−1)||_2^2 > ε then
                    Update u_j(l+1) by (16);
                    Exchange u_j(l+1) with other
                     actuators in the set A_i;
                    l ← l + 1;
                else
                    u_j(m) = u_j(l);
                end
                m ← m + 1;
            else
                u_j(k) = u_j(m);
            end
        end
    end
end
```

VI. Simulation Results

In this section, we consider 30 actuators and 100 sensors randomly deployed in a 150 m × 150 m ROI to monitor and control the temperature of 30 POIs. The values of the experimental set-up are described in Table I.

A. Performance Evaluation

Denote $\mathcal{J}_1(k) = \|\boldsymbol{A}\boldsymbol{x}(k|k) + \boldsymbol{B}\boldsymbol{u}(k) - \boldsymbol{x}^*\|_2^2$ and $\mathcal{J}_2(k) = \alpha\sum_{j=1}^{n_a} E_j(k)$ as the control error and the actuator energy consumption at the k^{th} step, respectively. Fig. 4(a) shows the dynamic change of the objective function $\mathcal{J}(k) = \mathcal{J}_1(k) + \mathcal{J}_2(k)$ using the proposed method. From it we can see that 1) the system is Bounded-Input-Bounded-Output (BIBO) stable, and 2) the proposed method satisfies the desired user's requirements and achieves a quick response to the events with a small energy consumption. Fig. 4(b) shows the system performance (objective function) to handle sequential events. Denote e_i as the event occurs at the system state x_i. Let $\mathcal{E}_1 = \{e_1, \ldots, e_{10}\}$, $\mathcal{E}_2 = \{e_{11}, \ldots, e_{20}\}$ and $\mathcal{E}_3 = \{e_{21}, \ldots, e_{30}\}$. Case 1 represents that the events $\mathcal{E}_1$, $\mathcal{E}_2$ and $\mathcal{E}_3$ occur at step $k = 1$ simultaneously. Case 2 and Case 3 represent that the events

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/JIOT.2018.2839030, IEEE Internet of Things Journal

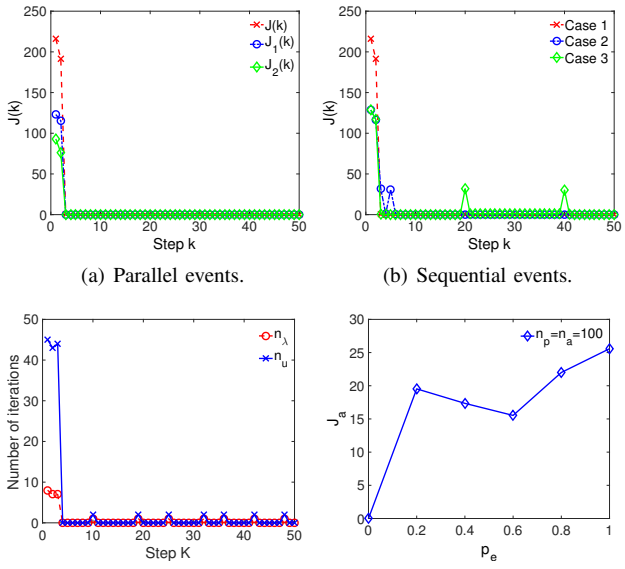
TABLE I. EXPERIMENTAL SET-UP

System model		
System matrix $\boldsymbol{A}$	$a_{ij} = \begin{cases} 1 - \frac{d_{ij}}{r_x} & \text{if } d_{ij} < r_x,\ i \neq j, \\ 0.9 & i = j, \\ 0 & \text{else.} \end{cases}$	
	d_{ij}: the distance between P_i and P_j	
	r_x: the impact range of system state	
Input matrix $\boldsymbol{B}$	$b_{ij} = \begin{cases} 1 - \frac{d_{ij}}{r_a} & \text{if } d_{ij} < r_a, \\ 0 & \text{else.} \end{cases}$	
	d_{ij}: the distance between P_i and A_j	
	r_a: the action range of actuator	
Measurement matrix $\boldsymbol{C}$	$c_{ij} = \begin{cases} 1 - \frac{d_{ij}}{r_s} & \text{if } d_{ij} < r_s, \\ 0 & \text{else.} \end{cases}$	
	d_{ij}: the distance between S_i and P_j	
	r_s: the sensing range of sensor	
$n_a = 30$	$n_s = 100$	$n_p = 30$
$r_x = 2$ m	$r_a = 20$ m	$r_s = 5$ m
$\alpha = 200$	$T_d = 2$ s	$\Delta = 0.5$ s
$\boldsymbol{x}^* = 25$ °C	$\boldsymbol{\epsilon} = 0.5$ °C	$\boldsymbol{x}(0) = 0$ °C
$\boldsymbol{Q} = 0.1\boldsymbol{I}$	$\boldsymbol{R} = 0.5\boldsymbol{I}$	$\varepsilon = 0.01$
$t^j_{i,m}(k) = \frac{k_t d_{ij} e_i(k)}{k_p \bar{u}_j^2}$	$T_{th} = \frac{\sum_{i=1}^{n_p} \max_{j=1}^{n_a} \{t^j_{i,m}(k)\}}{n_p} + T_d$	
	d_{ij}: the distance between P_i and A_j	
Actuator A_i characteristics		
$k_p = 1$	$[\underline{u}_j, \bar{u}_j] = [0, 50]$	$E^r_j(0) = 1000$ J
Control error $e_i(k)$ characteristics		
$k_t = 2$	$\eta_j \in [0, 0.2]$	
$t^j_i(k) = \frac{k_t d_{ij} e_i(k)}{p_j(k)}$	$\mathcal{G} = \sum_{j=1}^{n_p} \eta_j t^j_i(k)$	

$\mathcal{E}_1$, $\mathcal{E}_2$ and $\mathcal{E}_3$ occur at steps $k = 1, 3, 5$ and $k = 1, 20, 40$, respectively. In Case 1, the actuators take 4 steps to handle all the events. In Case 2, new events occur when the previous tasks haven't finished yet. Hence, the actuators' outputs change accordingly so as to match the characteristics of the occurring events. In Case 3, the actuators are in the idle state, when the assigned tasks are finished, and some actuators are activated again when new events occur. Fig. 4(a) and Fig. 4(b) show that the proposed method efficiently handles the parallel and sequential events.

The algorithm convergence (parallel events) is shown in Fig. 4(c), where $n_\lambda(k)$ and $n_u(k)$ are the number of outer-layer iterations and the total number of inner-layer iterations at the k^{th} step, respectively. In the beginning, the LDA requires at most eight times of outer-layer iterations, and at each outer-layer iteration, the actuator A_j only needs to exchange a small amount of information (i.e., its temporary output $u_j(l)$ six times in average). With the step k increasing, the outer-layer and inner-layer iterations numbers $n_\lambda(k)$ and $n_u(k)$ decrease very fast, since the objective function $\mathcal{J}(k)$ has converged. Hence, the communication overhead of the actuator A_j is small. For the sequential events, we obtain the similar results.

Random events (i.e., the events randomly occur at the system states $\{x_1, \ldots, x_{n_p}\}$) are very common in practice, especially without a priori knowledge about when and where the events will occur. In Fig. 4(d), we evaluate the system performance to handle random events. Denote $\mathcal{J}_a = \frac{\sum_{k=1}^{50} \mathcal{J}(k)}{50}$ as the average value of the objective function achieved by 50 steps, and n_e as the number of random events. We set $n_a = n_s = n_p = 100$, and the random events occur at step $k = 1$. The sensors, the actuators, and the POIs are randomly deployed in the ROI, and the event probability $p_e = \frac{n_e}{n_p}$ is changed from 0.2 to 1. Although $\mathcal{J}_a$ generally increases with p_e, the convergence speeds of LDA (i.e., the number of outer-layer iterations) under different p_e are almost the same.



(a) Parallel events. (b) Sequential events.

(c) Algorithm convergence. (d) Random events.

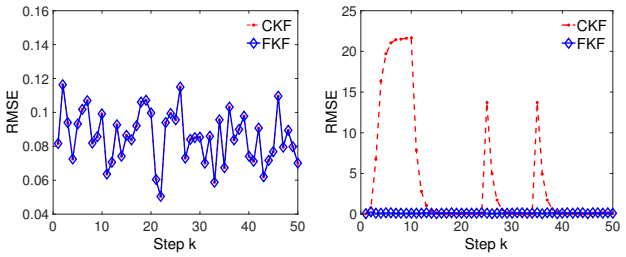
Fig. 4. System performance to handle parallel, sequential and random events.

B. Comparison with Existing Algorithms

In multi-sensor scenario, one of the popular methods to perform state estimation is the Centralized Kalman Filter (CKF) [37]. CKF gathers the measurements from all the sensors in one processing block (i.e., $\boldsymbol{z}(k) = [\boldsymbol{z}^1(k)', \ldots, \boldsymbol{z}^h(k)']'$), and runs KF to get the optimal estimation of system states. Fig. 5(a) compares the Root-Mean-Square Error (RMSE) of the estimation result achieved by CKF and FKF. Since that the RMSE of FKF and CKF are the same, and, thus, the FKF with data reset is an optimal state estimator. In WSANs, the data transmission between the sensors and the actuators are prone to time delay/packet loss, due to the limited computation/communication abilities of the sensors and the existence of the internal/external interferences [11]. If the unreliable sensors' measurements are used for state estimation, and they are undetected by both the MF and LFs, they will contaminate the fusion results. Since the measurement residuals $\boldsymbol{z}^j_i(k) - \boldsymbol{C}^j_i \boldsymbol{x}^j_i(k|k-1)$ of the j^{th} LF are Gaussian distributed, a simple χ^2 detector [40] can be employed to detect and isolate the faulty sensor measurement. We assume that if sensor S_j suffered from time delay/packet loss at step k, the measurement received by the actuator is $\boldsymbol{z}_j(k) = 0$. We randomly select 5 sensors to suffer from time delay/packet loss during the steps $k = 2 \sim 7, 25, 35$. The result is shown in Fig. 5(b). From it we can see that the estimation accuracy of CKF significantly drops. However, the estimation accuracy of FKF is almost not affected by the faulty measurements. Since CKF/FKF involves the multiplication and inversion of $n \times n$ matrices, the computational complexity is $O(n^3)$ [41]. For FKF and CKF, we have $n = D(\boldsymbol{z}^j_i)$ and $n = hD(\boldsymbol{z}^j_i)$, respectively. Although the CKF has a simple structure (i.e., no data fusion and reset), the distributed FKF-based S-A coordination achieves a better balance between the estimation accuracy and the computational complexity.

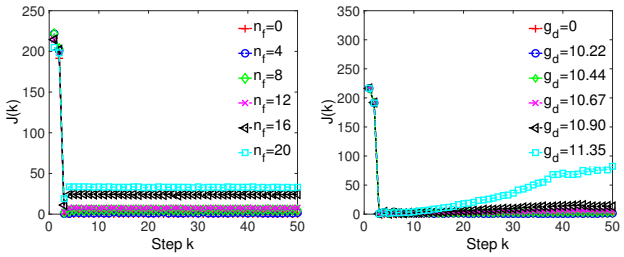
During the A-A coordination, we compare the proposed Lagrange-based Distributed Algorithm (LDA) with three ex-

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/JIOT.2018.2839030, IEEE Internet of Things Journal



(a) In normal conditions. (b) With faulty sensor measurements.

Fig. 5. Estimation performance comparison between CKF and FKF.



(a) Actuator failure. (b) Inaccurate system model.

Fig. 7. System performance under abnormal conditions.

isting methods: 1) PID Neural Network (PIDNN) [13], [30], 2) Gradient Descent method (GD) [7], [31], and 3) Sequential Unconstrained Minimization Technique (SUMT) [1]. The results are shown in Fig. 6. From it we can see that all four algorithms achieve the desired control accuracy. However, the error convergence speeds of LDA and SUMT are faster than that of PIDNN and GD. This is because: 1) PIDNN needs some steps to train the neural network in the beginning, and 2) LDA and SUMT minimize $\mathcal{J}(k)$ at each step k but PIDNN and GD minimize $\mathcal{J}(k)$ step by step. Since $\mathcal{J}(k)$ takes control error as well as energy consumption into account, a fast convergence speed implies a better real-time and energy-efficiency performance.

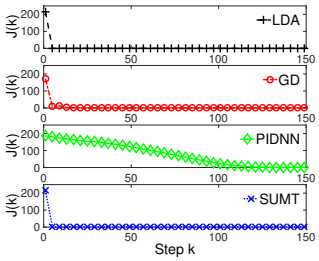


Fig. 6. System performance comparison between different control schemes.

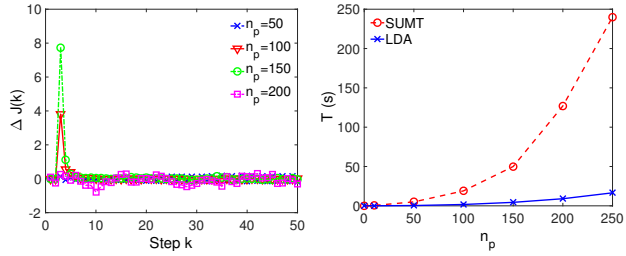
C. Robustness Analysis

The system model described by (1) has a general form (i.e., system matrix $\boldsymbol{A}$ and input matrix $\boldsymbol{B}$ are not limited to diagonal matrices). (1) also shows that the system states $\boldsymbol{x}(k)$ are controlled by the actuators' outputs $\boldsymbol{u}(k)$, and, thus, the actuator failure affects system performance. Denote n_f as the number of failing actuators. We assume that $u_j(k) = 0$, if the actuator A_j is out of function. As shown in Fig. 7(a), with n_f increasing, more and more POIs become uncontrollable. Hence, $\mathcal{J}(k)$ increases with n_f. On the other hand, since one POI may be covered by the multiple actuators, the failure of few actuators is still tolerable (e.g., $n_f = 4$).

From (16), we can see that the system performance depends on the system model accuracy. Fig. 7(b) shows the influence of inaccuracy affect of system model. Denote $g_d = \sum_{i=1}^{n_p}\sum_{j=1}^{n_p}|a_{ij}-\tilde{a}_{ij}|+\sum_{i=1}^{n_p}\sum_{j=1}^{n_a}|b_{ij}-\tilde{b}_{ij}|$ as a model error index, where $\{\boldsymbol{A},\boldsymbol{B}\}$ is the accurate system model used to simulate the dynamic of system, and $\{\tilde{\boldsymbol{A}},\tilde{\boldsymbol{B}}\}$ is the inaccurate system model used to calculate actuators' outputs $\boldsymbol{u}(k)$. When g_d is small, $\mathcal{J}(k)$ remains almost unchanged. However, for a large g_d, the system performance degrades or even becomes unstable. In order to achieve a desired control quality, a certain level of system model accuracy is essential. Therefore, we can use experimental/mechanism modeling [6], [7] or jointly estimate the system states $\boldsymbol{x}(k)$ and the system model $\{\boldsymbol{A},\boldsymbol{B}\}$ through the adaptive filtering methods [42], [43].

D. Scalability Evaluation

We compare the system performances achieved by LDA and SUMT under different network scales, with $n_a = n_p$ and $n_s = 100$. The sensors, the actuators, and the POIs are randomly deployed in the ROI so that the system partition is varied among the experiments. To show the difference between these two algorithms, we define a metric $\Delta\mathcal{J}(k) = \frac{\mathcal{J}(k)-\mathcal{J}_s(k)}{\mathcal{J}_s(0)} \times 100\%$, where $\mathcal{J}(k)$ and $\mathcal{J}_s(k)$ are the objective function values achieved by LDA and SUMT at the k^{th} step, respectively. Fig. 8 shows that 1) the performances of the two algorithms are quite similar, as the difference is less than 8%, and 2) with n_p increasing, the computing time of both algorithms grows. Since SUMT is an interior point method, the computational complexity is $O(n_a^3)$ [44]. On the other hand, since the inter-layer and outer-layer iterations of LDA are based on subgradient method, the computational complexity is $O(U^2R^2\varepsilon^{-2})$ [45], where U is the distance between an optimal solution and the initial point and R is a Lipschitz constant for the objective function. Hence, the LDA has much shorter computing time than the SUMT. Although the centralized method is able to collect the global information and achieve a globally optimal control decision, the communication latency is also large, since the sensory data and the control command are routed through the sink rather than the nearby nodes. The communication and the computation overhead at the sink and at the nodes near to the sink increase significantly with the network scale.



(a) Objective function. (b) Computing time.

Fig. 8. Scalability comparison between SUMT and LDA.

Fig. 9 evaluates the system performance under different node configurations. We consider following two cases: 1) $n_p = 50$ and n_a changes from 50 to 100; 2) $n_a = 50$ and n_p changes from 50 to 100. The sensors, the actuators, and the POIs are randomly deployed in the ROI. In Case 1, with n_a decreasing, $\mathcal{J}_a$ increases. This is because the more actuators are involved

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/JIOT.2018.2839030, IEEE Internet of Things Journal

in event processing, the easier is to handle the events. In Case 2, with n_p increasing, $\mathcal{J}_a$ increases. This is because 1) each actuator only covers a limited area in the ROI, and 2) the POIs are randomly deployed. With n_p increasing, some POIs are out of actuators' control, which leads to a large control error.

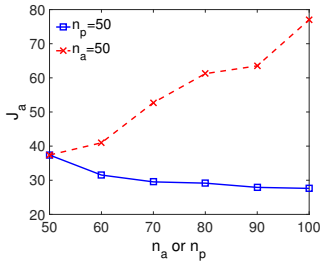


Fig. 9. System performance under different node configurations.

VII. Conclusion

The distributed node coordination problem in WSANs is addressed in this paper. We aim at satisfying the user's requirements on the system states in a real-time and energy-efficient manner. The node coordination problem was formulated as a non-linear programming, which was proved to be convex. We applied a distributed estimation method, which is based on Voronoi cells and FKF, to perform S-A coordination. Based on the estimated result, we proposed a Lagrange-based distributed algorithm to adjust actuators' outputs and proved its convergence. Simulation results demonstrated the performance of the proposed method in terms of the quality of the solution and the computing time.

Acknowledgment

This research was partly supported by the National Natural Science foundation of China (Grant No. 61403340 and 61573103), the INRIA post-doctoral research fellowship program, the State Key Laboratory of Synthetical Automation for Process Industries, and the Fundamental Research Funds for the Central Universities.

Appendix A
Proof of Theorem 3.1

Proof: From $E_j(k) = \Delta p_j(k) = \Delta k_p u_j(k)^2$, we have $\frac{\partial^2 E_j(k)}{\partial u_j(k)^2} = 2\Delta k_p$, $\frac{\partial^2 E_j(k)}{\partial u_i(k)^2} = 0$, $\frac{\partial^2 E_j(k)}{\partial u_j(k)\partial u_l(k)} = 0$ $(l \neq j)$. And further, $\nabla^2(\sum_{j=1}^{n_a} E_j(k)) = \sum_{j=1}^{n_a} \nabla^2 E_j(k) = 2\Delta k_p \boldsymbol{I}_{n_a \times n_a}$. Hence, the Hessian of $E_j(k)$ and $\sum_{j=1}^{n_a} E_j(k)$ are positive semi-definite (i.e., $E_j(k)$ and $\sum_{j=1}^{n_a} E_j(k)$ are convex). Note that the second norm is convex (e.g., $\|\boldsymbol{A}\boldsymbol{x}(k|k) + \boldsymbol{B}\boldsymbol{u}(k) - \boldsymbol{x}^*\|_2^2$), and the sum operation preserves the convexity. The objective function of the primal problem is convex. Let $\mathcal{G}(k) = \mathcal{G}(e_i(k), \mathcal{U}_i(k))$. Since $\frac{\partial^2 \mathcal{G}(k)}{\partial u_j(k)^2} \geq 0$ and $\frac{\partial^2 \mathcal{G}(k)}{\partial u_j(k)\partial u_l(k)} = 0$ $(l \neq j)$, the Hessian of $T_i(k)$ is positive semi-definite. Hence, the constraints of the primal problem are convex. According to the definition of convex problem [44], the PP (3) is a convex optimization problem. ■

Appendix B
Proof of Theorem 5.1

Proof: Let $\mathcal{H}_j(m) = \frac{\partial \mathcal{R}(\boldsymbol{\lambda}(m))}{\partial \lambda_j(m)}$. Therefore, $\boldsymbol{\mathcal{H}}(m) = [\mathcal{H}_1(m), \ldots, \mathcal{H}_{n_a+n_p}(m)]'$ is the subgradient vector of dual function $\mathcal{R}$. We construct a Lyapunov function

$$\mathcal{V}(\boldsymbol{\lambda}(m)) = \sum\nolimits_{j=1}^{n_a+n_p} (\lambda_j^* - \lambda_j(m))^2.$$

According to $\lambda_j(m+1) = [\lambda_j(m) + \delta\mathcal{H}_j(m)]^+$, we have $\lambda_j(m+1) \geq 0$ and $\lambda_j(m) \geq 0$. Since $\delta > 0$ and $\mathcal{H}_j(m) \leq 0$, we get $\lambda_j(m+1) \geq \lambda_j(m) + \delta\mathcal{H}_j(m)$. And further, due to $\lambda_j^* \geq 0$, we have $(\lambda_j^* - \lambda_j(m+1))^2 \leq (\lambda_j^* - \lambda_j(m) - \delta\mathcal{H}_j(m))^2$. Based on this inequality, we obtain

$$\begin{aligned}
&\mathcal{V}(\boldsymbol{\lambda}(m+1)) = \sum\nolimits_{j=1}^{n_a+n_p} (\lambda_j^* - \lambda_j(m+1))^2 \\
\leq &\sum\nolimits_{j=1}^{n_a+n_p} (\lambda_j^* - \lambda_j(m) - \delta\mathcal{H}_j(m))^2 \\
= &\mathcal{V}(\boldsymbol{\lambda}(m)) + \sum\nolimits_{j=1}^{n_a+n_p} 2\delta\mathcal{H}_j(m)(\lambda_j(m) - \lambda_j^*) \\
&+ \sum\nolimits_{j=1}^{n_a+n_p} \delta^2 \mathcal{H}_j(m)^2. \qquad (18)
\end{aligned}$$

Note that $\mathcal{R}(\boldsymbol{\lambda})$ is concave. According to the definition of subgradient, we get

$$\sum\nolimits_{j=1}^{n_a+n_p} \mathcal{H}_j(m)(\lambda_j(m) - \lambda_j^*) \leq \mathcal{R}(\boldsymbol{\lambda}(m)) - \mathcal{R}(\boldsymbol{\lambda}^*). \quad (19)$$

Substituting (19) into (18) and applying the inequalities recursively, we have

$$\begin{aligned}
&\mathcal{V}(\boldsymbol{\lambda}(m+1)) \\
\leq &\mathcal{V}(\boldsymbol{\lambda}(m)) + 2\delta(\mathcal{R}(\boldsymbol{\lambda}(m)) - \mathcal{R}(\boldsymbol{\lambda}^*)) + \sum\nolimits_{j=1}^{n_a+n_p} \delta^2 \mathcal{H}_j(m)^2 \\
\leq &\mathcal{V}(\boldsymbol{\lambda}(1)) + \sum\nolimits_{\tau=1}^{m} 2\delta(\mathcal{R}(\boldsymbol{\lambda}(\tau)) - \mathcal{R}(\boldsymbol{\lambda}^*)) \\
&+ \sum\nolimits_{\tau=1}^{m} \sum\nolimits_{j=1}^{n_a+n_p} \delta^2 \mathcal{H}_j(\tau)^2. \qquad (20)
\end{aligned}$$

Let $\overline{\boldsymbol{\lambda}}(m) = \frac{1}{m}\sum_{\tau=1}^{m} \boldsymbol{\lambda}(\tau)$. Utilizing the concavity of $\mathcal{R}(\boldsymbol{\lambda})$ and Jensen's inequality, we get

$$\sum\nolimits_{\tau=1}^{m} (\mathcal{R}(\boldsymbol{\lambda}(\tau)) - \mathcal{R}(\boldsymbol{\lambda}^*)) \leq m(\mathcal{R}(\overline{\boldsymbol{\lambda}}(m)) - \mathcal{R}(\boldsymbol{\lambda}^*)). \quad (21)$$

Assume that $\sum_{j=1}^{n_a+n_p} \mathcal{H}_j(\tau)^2 \leq \mathcal{B}$ since $\boldsymbol{\mathcal{H}}(\tau)$ is bounded. Substituting (21) into (20) and noting that $\mathcal{V}(\boldsymbol{\lambda}(m+1)) \geq 0$, we have

$$\mathcal{R}(\boldsymbol{\lambda}^*) - \mathcal{R}(\overline{\boldsymbol{\lambda}}(m)) \leq \frac{\mathcal{V}(\boldsymbol{\lambda}(1)) + m\delta^2\mathcal{B}}{2m\delta}.$$

Hence, we get $\limsup_{m\to\infty} \left(\mathcal{R}(\boldsymbol{\lambda}^*) - \mathcal{R}(\overline{\boldsymbol{\lambda}}(m))\right) \leq \frac{\delta\mathcal{B}}{2}$. According to the definition of statistical convergence, given a small enough δ, $\boldsymbol{\lambda}$ statistically converges to $\boldsymbol{\lambda}^*$. ■

Appendix C
Proof of Theorem 5.2

Proof: We construct a Lyapunov function

$$\mathcal{T}(\boldsymbol{\lambda}(m)) = \sum\nolimits_{j=1}^{n_a+n_p} \lambda_j(m)^2.$$

Since $\lambda_j(m+1)^2 \leq (\lambda_j(m) + \delta\mathcal{H}_j(m))^2$, we have

$$\mathcal{T}(\boldsymbol{\lambda}(m+1)) = \sum\nolimits_{j=1}^{n_a+n_p} \lambda_j(m+1)^2$$

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/JIOT.2018.2839030, IEEE Internet of Things Journal

$$
\begin{aligned}
&\leq \sum\nolimits_{j=1}^{n_a+n_p} (\lambda_j(m) + \delta \mathcal{H}_j(m))^2 \\
&= \mathcal{T}(\boldsymbol{\lambda}(m)) + \sum\nolimits_{j=1}^{n_a+n_p} (2\delta\lambda_j(m)\mathcal{H}_j(m) + \delta^2 \mathcal{H}_j(m)^2) \\
&= \mathcal{T}(\boldsymbol{\lambda}(m)) + 2\delta(\mathcal{J}(\boldsymbol{u}(m)) + \sum\nolimits_{j=1}^{n_a+n_p} \lambda_j(m)\mathcal{H}_j(m)) \\
&\quad - 2\delta\mathcal{J}(\boldsymbol{u}(m)) + \sum\nolimits_{j=1}^{n_a+n_p} \delta^2 \mathcal{H}_j(m)^2 \\
&\leq \mathcal{T}(\boldsymbol{\lambda}(m)) + 2\delta(\mathcal{J}(\boldsymbol{u}^*) + \sum\nolimits_{j=1}^{n_a+n_p} \lambda_j(m)\mathcal{H}_j(\boldsymbol{u}^*)) \\
&\quad - 2\delta\mathcal{J}(\boldsymbol{u}(m)) + \sum\nolimits_{j=1}^{n_a+n_p} \delta^2 \mathcal{H}_j(m)^2 \qquad (22a) \\
&\leq \mathcal{T}(\boldsymbol{\lambda}(m)) + 2\delta(\mathcal{J}(\boldsymbol{u}^*) - \mathcal{J}(\boldsymbol{u}(m))) \\
&\quad + \sum\nolimits_{j=1}^{n_a+n_p} \delta^2 \mathcal{H}_j(m)^2 \qquad (22b) \\
&\leq \mathcal{T}(\boldsymbol{\lambda}(1)) + \sum\nolimits_{\tau=1}^{m} 2\delta(\mathcal{J}(\boldsymbol{u}^*) - \mathcal{J}(\boldsymbol{u}(\tau))) \\
&\quad + \sum\nolimits_{\tau=1}^{m} \sum\nolimits_{j=1}^{n_a+n_p} \delta^2 \mathcal{H}_j(\tau)^2, \qquad (22c)
\end{aligned}
$$

where $\mathcal{H}_j(\boldsymbol{u}^*)$ represents substituting $\boldsymbol{u}^*$ into $\frac{\partial \mathcal{R}_i(\boldsymbol{\lambda}_i(m))}{\partial \lambda_j(m)}$. (22a) holds since $\boldsymbol{u}(m)$ is the minimizer in the problem (15) under the given $\boldsymbol{\lambda}(m)$, and (22b) holds since since $\delta \geq 0$, $\lambda_j(m) \geq 0$ and $\mathcal{H}_j(\boldsymbol{u}^*) \leq 0$.

Utilizing the convexity of $\mathcal{J}(\boldsymbol{u})$ and Jensen's inequality, we get

$$\sum\nolimits_{\tau=1}^{m} (\mathcal{J}(\boldsymbol{u}^*) - \mathcal{J}(\boldsymbol{u}(\tau))) \leq m(\mathcal{J}(\boldsymbol{u}^*) - \mathcal{J}(\overline{\boldsymbol{u}}(m))), \quad (23)$$

where $\overline{\boldsymbol{u}}(m) = \frac{1}{m}\sum_{\tau=1}^{m} \boldsymbol{u}(\tau)$. Substituting (23) into (22c) and noting that $\mathcal{T}(\boldsymbol{u}(m+1)) \geq 0$, we have

$$\mathcal{J}(\overline{\boldsymbol{u}}(m)) - \mathcal{J}(\boldsymbol{u}^*) \leq \frac{\mathcal{T}(\boldsymbol{\lambda}(1)) + m\delta^2\mathcal{B}}{2m\delta}.$$

From $\limsup_{m\to\infty}(\mathcal{J}(\overline{\boldsymbol{u}}(m)) - \mathcal{J}(\boldsymbol{u}^*)) \leq \frac{\delta\mathcal{B}}{2}$, we can get that $\boldsymbol{u}$ statistically converges to $\boldsymbol{u}^*$ when δ is a small enough value. ■

REFERENCES

[1] L. Mo, X. Cao, J. Chen, and Y. Sun, "Collaborative estimation and actuation for wireless sensor and actuator networks," in *Proc. IFAC World Congress*, 2014, pp. 5544–5549.

[2] R. Deng, S. He, P. Cheng, and Y. Sun, "Towards balanced energy charging and transmission collision in wireless rechargeable sensor networks," *J. Commun. Netw.*, vol. 19, no. 4, pp. 341–350, 2017.

[3] X. Cao, L. Liu, Y. Cheng, and X. S. Shen, "Towards energy-efficient wireless networking in the big data era: A survey," *IEEE Commun. Surv. Tut.*, vol. 20, no. 1, pp. 303–332, 2018.

[4] Y. Liu, Y. Peng, B. Wang, S. Yao, and Z. Liu, "Review on cyber-physical systems," *IEEE/CAA J. of Autom. Sinica*, vol. 4, no. 1, pp. 27–40, 2017.

[5] J. Chen, K. Hu, Q. Wang, Y. Sun, Z. Shi, and S. He, "Narrowband internet of things: Implementations and applications," *IEEE Internet Things J.*, vol. 4, no. 6, pp. 2309–2314, 2017.

[6] L. Yeh, C. Lu, C. Kou, Y. Tseng, and C. Yi, "Autonomous light control by wireless sensor and actuator networks," *IEEE Sensors J.*, vol. 10, no. 6, pp. 1029–1041, 2010.

[7] X. Cao, J. Chen, Y. Xiao, and Y. Sun, "Building-environment control with wireless sensor and actuator networks: centralized versus distributed," *IEEE Trans. Ind. Electron.*, vol. 57, no. 11, pp. 3596–3605, 2010.

[8] K. Ota, M. Dong, Z. Cheng, J. Wang, X. Li, and X. S. Shen, "ORACLE: Mobility control in wireless sensor and actor networks," *Comput. Commun.*, vol. 35, no. 9, pp. 1029–1037, 2012.

[9] X. Luo, L. Feng, J. Yan, and X. Guan, "Dynamic coverage with wireless sensor and actor networks in underwater environment," *IEEE/CAA J. of Autom. Sinica*, vol. 2, no. 3, pp. 274–281, 2015.

[10] J. Chen, X. Cao, P. Cheng, Y. Xiao, and Y. Sun, "Distributed collaborative control for industrial automation with wireless sensor and actuator networks," *IEEE Trans. Ind. Electron.*, vol. 57, no. 12, pp. 4219–4230, 2010.

[11] H. Li, L. Lai, and H. V. Poor, "Multicast routing for decentralized control of cyber physical systems with an application in smart grid," *IEEE J. Sel. Areas Commun.*, vol. 30, no. 6, pp. 1097–1107, 2012.

[12] Y. Zhang, R. Yu, S. Xie, W. Yao, Y. Xiao, and M. Guizani, "Home M2M networks: Architectures, standards, and QoS improvement," *IEEE Commun. Mag.*, vol. 49, no. 4, pp. 44–52, 2011.

[13] L. Mo, B. Xu, X. You, and L. Liu, "Distributed lighting system based on wireless sensor and actuator networks," *Ad Hoc & Sensor Wireless Netw.*, vol. 20, no. 2, pp. 21–46, 2014.

[14] T. Melodia, D. Pompili, V. C. Gungor, and I. F. Akyildiz, "Communication and coordination in wireless sensor and actor networks," *IEEE Trans. Mobile Comput.*, vol. 6, no. 10, pp. 1116–1129, 2007.

[15] K. Gatsis, A. Ribeiro, and G. J. Pappas, "Optimal power management in wireless control systems," *IEEE Trans. Autom. Control*, vol. 59, no. 6, pp. 1495–1510, 2014.

[16] V. Gungor, O. Akan, and I. Akyildiz, "A real-time and reliable transport protocol for wireless sensor and actor networks," *IEEE/ACM Trans. Netw.*, vol. 16, no. 2, pp. 359–370, 2008.

[17] E. Ngai, Y. Zhou, M. R. Lyu, and J. Liu, "A delay-aware reliable event reporting framework for wireless sensor–actuator networks," *Ad Hoc Netw.*, vol. 8, no. 7, pp. 694–707, 2010.

[18] C. Farias, L. Pirmez, F. Delicato, L. Carmo, W. Li, A. Zomaya, and J. De Souza, "Multisensor data fusion in shared sensor and actuator networks," in *Proc. IEEE FUSION*, 2014, pp. 1–8.

[19] F. Cattivelli and A. Sayed, "Diffusion strategies for distributed Kalman filtering and smoothing," *IEEE Trans. Autom. Control*, vol. 55, no. 9, pp. 2069–2084, 2010.

[20] S. Das and J. Moura, "Distributed Kalman filtering with dynamic observations consensus," *IEEE Trans. Signal Process.*, vol. 63, no. 17, pp. 4458–4473, 2015.

[21] R. Yu, Y. Zhang, K. Yang, S. Xie, and H. H. Chen, "Distributed geographical packet forwarding in wireless sensor and actuator networks - a stochastic optimal control approach," *IET Wireless Sensor Systems*, vol. 2, no. 1, pp. 63–74, 2012.

[22] T. Melodia, D. Pompili, and I. F. Akyldiz, "Handling mobility in wireless sensor and actor networks," *IEEE Trans. Mobile Comput.*, vol. 9, no. 2, pp. 160–173, 2010.

[23] Y. Zhang, R. Yu, M. Nekovee, Y. Liu, S. Xie, and S. Gjessing, "Cognitive machine-to-machine communications: visions and potentials for the smart grid," *IEEE Netw.*, vol. 26, no. 3, pp. 6–13, 2012.

[24] M. Mazo and P. Tabuada, "Decentralized event-triggered control over wireless sensor/actuator networks," *IEEE Trans. Autom. Control*, vol. 56, no. 10, pp. 2456–2461, 2011.

[25] X. Cao, P. Cheng, J. Chen, and Y. Sun, "An online optimization approach for control and communication codesign in networked cyber-physical systems," *IEEE Trans. Ind. Electron.*, vol. 9, no. 1, pp. 439–450, 2013.

[26] M. Dong, K. Ota, and A. Liu, "RMER: Reliable and energy-efficient data collection for large-scale wireless sensor networks," *IEEE Internet Things J.*, vol. 3, no. 4, pp. 511–519, 2016.

[27] G. Xie, K. Ota, M. Dong, F. Pan, and A. Liu, "Energy-efficient routing for mobile data collectors in wireless sensor networks with obstacles," *Peer-to-Peer Netw. Appl.*, vol. 10, no. 3, pp. 472–483, 2017.

[28] M. Okhovvat, M. Sharifi, and H. Momeni, "Task allocation to actors in wireless sensor actor networks: an energy and time aware technique," *Procedia Comput. Sci.*, vol. 3, pp. 484–490, 2011.

[29] A. D'Innocenzo, M. D. D. Benedetto, and E. Serra, "Fault tolerant control of multi-hop control networks," *IEEE Trans. Autom. Control*, vol. 58, no. 6, pp. 1377–1389, 2013.

[30] L. Mo and B. Xu, "Coordination mechanism based on mobile actuator design for wireless sensor and actuator networks," *Wirel. Commun. Mob. Comput.*, vol. 15, pp. 1274–1289, 2015.

[31] M. Nakamura, A. Sakurai, and J. Nakamura, "Distributed environment control using wireless sensor/actuator networks for lighting applications," *Sensors*, vol. 9, no. 11, pp. 8593–8609, 2009.

[32] C. Wu, G. S. Tewolde, W. Sheng, B. Xu, and Y. Wang, "Distributed multi-actuator control for workload balancing in wireless sensor and

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/JIOT.2018.2839030, IEEE Internet of Things Journal

actuator networks," *IEEE Trans. Autom. Control*, vol. 56, no. 10, pp. 2462–2467, 2011.

[33] Y. Wu, Q. Chaudhari, and E. Serpedin, "Clock synchronization of wireless sensor networks," *IEEE Signal Process. Mag.*, vol. 28, no. 1, pp. 124–138, 2011.

[34] Q. Ge, T. Shao, Z. Duan, and C. Wen, "Performance analysis of the Kalman filter with mismatched noise covariances," *IEEE Trans. Autom. Control*, vol. 61, no. 12, pp. 4014–4019, 2016.

[35] N. Carlson, "Federated square root filter for decentralized parallel processors," *IEEE Trans. Aerosp. Electron. Syst.*, vol. 26, no. 3, pp. 517–525, 1990.

[36] A. Bertrand and M. Moonen, "Seeing the bigger picture: How nodes can learn their place within a complex Ad Hoc network topology," *IEEE Signal Process. Mag.*, vol. 30, no. 3, pp. 71–82, 2013.

[37] Y. S. Kim and K. S. Hong, "Decentralized information filter in federated form," in *Proc. IEEE SICE*, 2003, pp. 2176–2181.

[38] R. Deng, Y. Zhang, S. He, J. Chen, and X. Shen, "Maximizing network utility of rechargeable sensor networks with spatiotemporally coupled constraints," *IEEE J. Sel. Areas Commun.*, vol. 34, no. 5, pp. 1307–1319, 2016.

[39] L. Chen, S. Low, M. Chiang, and J. Doyle, "Cross-layer congestion control, routing and scheduling design in ad hoc wireless networks," in *Proc. IEEE INFOCOM*, 2006, pp. 1–13.

[40] Y. Mo, R. Chabukswar, and B. Sinopoli, "Detecting integrity attacks on scada systems," *IEEE Trans. Control Syst. Technol.*, vol. 22, no. 4, pp. 1396–1407, 2014.

[41] U. A. Khan and J. M. F. Moura, "Distributing the Kalman filter for large-scale systems," *IEEE Trans. Signal Process.*, vol. 56, no. 10, pp. 4919–4935, 2008.

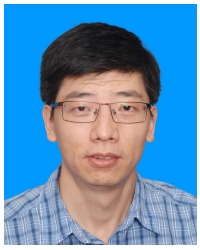
[42] A. G. Parlos, S. K. Menon, and A. Atiya, "An algorithmic approach to adaptive state filtering using recurrent neural networks," *IEEE Trans. Neural Netw.*, vol. 12, no. 6, pp. 1411–1432, 2001.

[43] L. Mo and B. Xu, "Adaptive filtering based collaborative actuation for wireless sensor and actuator networks," *Int. J. Ad Hoc Ubiq. Co.*, vol. 20, no. 4, pp. 223–236, 2015.

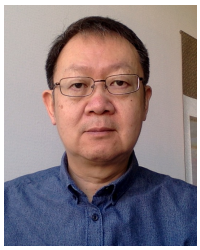
[44] S. Joshi and S. Boyd, "Sensor selection via convex optimization," *IEEE Trans. Signal Process.*, vol. 57, no. 2, pp. 451–462, 2009.

[45] M. Ito and M. Fukuda, "A family of subgradient-based methods for convex optimization problems in a unifying framework," *Optimization Methods and Software*, vol. 31, no. 5, pp. 952–982, 2016.

Lei Mo (S'13–M'17) is currently a Postdoctoral Fellow with INRIA Rennes Bretagne Atlantique research center, France. He received the B.S. degree from College of Telecom Engineering and Information Engineering, Lanzhou University of Technology, Lanzhou, China, in 2007, and the Ph.D. degree from College of Automation Science and Engineering, South China University of Technology, Guangzhou, China, in 2013. From 2013 to 2015, he was a research fellow with the Department of Control Science and Engineering, Zhejiang University, China. From 2015 to 2017, he was a research fellow with INRIA Nancy Grand Est research center, France. His current research interests include networked estimation and control in wireless sensor and actuator networks, cyber-physical systems, task mapping and resources allocation in multi-core systems. He serves/served as a Guest Editor for IEEE Access and Journal of Computer Networks and Communications. He also serves/served as a TPC Member for WF-5G 2018, Glebocom-MWN 2018 and COMNETSAT 2018.



Xianghui Cao (S'08–M'11–SM'16) received the B.S. and Ph.D. degrees in control science and engineering from Zhejiang University, Hangzhou, China, in 2006 and 2011, respectively. From 2007 to 2009, he was a visiting scholar with the Department of Computer Science, University of Alabama, USA. From 2012 to 2015, he was a senior research associate with the Department of Electrical and Computer Engineering, Illinois Institute of Technology, USA. He is currently an associate professor with the School of Automation, Southeast University, Nanjing, China. His research interests include cyber-physical systems, wireless network performance analysis, wireless networked control, and network security. He serves as the Publicity Co-Chair for ACM MobiHoc 2015, Symposium Co-Chair for ICNC 2017 and IEEE/CIC ICCC 2015, and a TPC Member for a number of conferences. He also serves as an Associate Editor for several journals, including ACTA Automatica Sinica, IEEE/CAA Journal of Automatica Sinica, KSII Transactions on Internet and Information Systems, Security and Communication Networks, and International Journal of Ad Hoc and Ubiquitous Computing. He was a recipient of the Best Paper Runner-Up Award from ACM MobiHoc 2014.



Yeqiong Song received the B.S. degree from Beijing University of Posts and Telecommunications, Beijing, China, in 1984, the Master degree from the Ecole National Supérieure des Telecommuications (ENST), Paris, France, in 1987, the M.Sc. degree (DEA) from the University of Paris 6 in 1988, the Ph.D. degree from the Institut National Polytechnique de Lorraine (INPL), Nancy, France, in 1991, and the Habilitation to lead research (HdR) degree from the University of Henri Poincaré Nancy 1 in 2004, all in telecommunications and computer science. In 1988, he joined Lorraine Laboratory of IT Research and Applications (LORIA) in Nancy, France. From 1992 to 2005, he was an Associate Professor with the University of Henri Poincare Nancy 1. From 2001 to 2003, he was a Full-Time Researcher with INRIA Lorraine. Since 2005, he is a full Professor of computer science with the INPL and now with the University of Lorraine. His research interests include modeling and performance evaluation of networks and real-time distributed systems including networked control systems, and the implementation of real-time quality-of-service mechanisms in industrial networks, in-vehicle networks, and wireless IoT and sensor networks. He authored/co-authored more than 100 scientific papers.

Angeliki Kritikakou is currently an Associate Professor at University of Rennes 1 and IRISA - INRIA Rennes Bretagne Atlantique research center. She received her Ph.D. in 2013 from the Department of Electrical and Computer Engineering at University of Patras, Greece and in collaboration with IMEC Research Center, Belgium. She worked for one year as a Postdoctoral Research Fellow at the Department of Modelling and Information Processing (DTIM) at ONERA in collaboration with Laboratory of Analysis and Architecture of Systems (LAAS) and the University of Toulouse, France. Her research interests include embedded systems, real-time systems, mixed-critical systems, hardware/software co-design, mapping methodologies, design space exploration methodologies, memory management methodologies, low power design and fault tolerance.